A Binary Complete and Aperiodic Turing machine

Pablo Concha-Vega, Rodrigo Torres-Avilés

A Binary Complete and Aperiodic Turing machine

Pablo Concha-Vega[1], Rodrigo Torres-Avilés[2⋆]

[1] *Dpto. de Ciencias de la Computación y Tecnologias de la Información, Universidad de Bío-Bío, Chile*

[2] *Dpto. de Sistemas de Información, Universidad del Bío-Bío, Chile*

Received 04 August 2020; In final form XX xxxxx 202X

Turing machines have been studied as dynamical systems for more than two decades, first formalized by Kůrka, proposing a topological dynamical system named *Turing machine with moving tape* (TMT). It was conjectured that every TMT has at least one periodic point. Nowadays, there are several examples of aperiodic Turing machines, disproving Kůrka's conjecture. Moreover, one of these machines, named SMART, has other interesting properties like reversibility, completeness, aperiodicity, topological minimality, among others. This machine has four states and works over an alphabet of three symbols. In this research, we study the dynamical properties of *BinSmart*, a 2-symbols reconstruction of the main dynamic of SMART machine. This machine results to be aperiodic, topologically minimal (therefore transitive) but not time-symmetric, as it is not a direct translation of the original machine. We also prove that its t-shift is a primitive substitution.

Key words: Symbolic Systems, Topological Dynamics, Turing machines, Subshifts, Minimality, Aperiodicity, Substitution

1 INTRODUCTION

The term *computation* refers to any kind of mathematical calculation that includes both arithmetical and non-arithmetical steps and follows

⋆ email: rtorres@ubiobio.cl

a well-defined model, for example, an algorithm. While algorithms had an important role in certain areas of mathematics, prior to the 1930's they had not been studied as mathematical objects by themselves. [11] changed this by introducing a type of imaginary machine which could take an input and process it with a finite number of steps until getting the final output. This mathematical model of computation defines an abstract machine, which classically consists of a head that, following some defined instructions, can manipulate symbols and move along an infinite tape (but with a finite input). The part of the tape that is not part of the input is filled with a special symbol usually known as the *blank symbol*. This model was developed as a formalization of the concept of computation.

The study of Turing machines is mainly focused on computability, which in simple words means to study whether computers can solve a certain problem. One of the most studied problems related with this topic is the *halting problem*, which consists of determining, given a computer program and an input, whether the program will finish running or will be running forever. Alan Turing proved in 1936 that an algorithm to solve the halting problem for all possible inputs (A program and an input for this program) cannot exist.

Nevertheless, in this research, we do not study neither the classical Turing machine model or computability and complexity. Instead, we consider Turing machines as *symbolic systems*, more specifically, a model defined by [8] in 1997 named *Turing machine with moving tape* (TMT), which, as its name suggests, is the tape the component that has the ability to move while the head keeps stationary. It is important to remark that if the model is not defined in this way, it could not be compact, which is a serious drawback in topological dynamics [8]. The idea behind this model is to study the dynamics of Turing machines, so we do not consider either initial or final states, since the computation may start in any state and proceed infinitely. We do not need to restrict the computation only to finite tape contents either, because the computation is carried out over an arbitrarily long part of the tape. Simply stated, the Turing machine with moving tape model can be defined with a finite set of inner states, a finite alphabet, and a transition function.

Considering this dynamical point of view, interesting questions emerge when we work over infinite 'inputs' or infinite computational time. Can we predict the behavior of the computer? Can we know if the computer

will reach a specific state of computation? Can we determine if the computation will fall into a loop? If it does not, will it reach any possible state of computation? How much can the dynamical point of view tell us about computation? There exists a very rich field of research attaining to these questions (for examples, see [6, 5, 4, 10]).

The study of aperiodic Turing machines has been of particular interest. Aperiodicity means that there does not exist a configuration in the machine that eventually evolves into itself. In the context of TMT, Kůrka conjectured that there does not exist such a Turing machine. This was later disproved by [2], presenting an aperiodic, complete, but irreversible Turing machine. Several years later, [3] presented a 3-symbol, aperiodic and complete, but this time reversible, Turing machine called SMART, proving a conjecture by [7] about its existence. In this last work, it was also proved that SMART presented several other dynamical properties not seen before in TMT, such as Topological Minimality and Substitutivity. This machine was also used to prove that aperiodicy and minimality are undecidable properties.

In topological dynamics, even a change in the speed of the machine could imply changes in its dynamics and properties. Considering the previous, a simple decrease in the size of the SMART alphabet would not give us a binary machine with all of its topological properties (in fact, a naive transformation would imply breaking the reversibility or completeness of the Turing machine). In this fashion, the existence of a two symbol complete reversible and aperiodic Turing machine remains unclear.

In this research, we focus on the study of a Turing machine proposed by Julien Cassaigne, nicknamed *BinSmart*. This Turing machine is claimed to be aperiodic, based in the same general behavior of SMART: recursive calls of searches for a particular symbol. Therefore, the research question is to answer which properties are preserved from SMART to BinSmart.

This research is organized as follows: In section 2 all the main concepts needed for the rest of the document are defined. Next, in section 3, the dynamical and topological properties of BinSmart are proved. Last, in section 4, a substitution for the minimal t-shift of BinSmart is presented.

2 DEFINITIONS

2.1 Dynamical System

A *dynamical system* is a pair (X, T), where X is a compact metric space and $T : X \to X$ is a continuous self-map called *global transition function*. The compact metric space evolves in time through the global transition function, thus the $n-$th evolution of an element $x \in X$ is denoted by $T^n(x)$. When the dynamical system works over discrete time it is called *discrete dynamical system*, and when the metric space is defined in a discrete way, it is called a *symbolic system*.

2.2 Orbits

In a dynamical system (X, T), the orbit $\mathcal{O}(x)$ of a point $x \in X$ is defined by $\mathcal{O}(x) = (T^n(x))_{n\in\mathbb{N}}$, in other words, the orbit of a point contains all its evolutions.

Periodic orbits: A point $x \in X$ has a *periodic orbit* if there exists $n \in \mathbb{N}$ such that $x = T^n(x)$, or equivalently, $x \in \mathcal{O}(x)$. The point $x \in X$ is called *periodic point* with period n. If the dynamical system does not possess a periodic point, it is called *Aperiodic*.

2.3 Subshifts, languages and words

There exists a specific type of symbolic system called *subshift*, which is based in a space of words evolving through the shift function. To give a formal definition, we need first some concepts about words.

Given a finite set Σ, called *alphabet*, $\Sigma^{\mathbb{Z}}$ is the set of bi-infinite sequences of elements of Σ, called *bi-infinite words*. Σ^{ω} ($^{\omega}\Sigma$) represent the set of right (left) infinite sequences of elements of Σ, called *infinite words to the right (left)*. The set of infinite words to the right can be also represented by $\Sigma^{\mathbb{N}}$. Finally, Σ^* represents the set of finite concatenations of elements of Σ, called *finite words*, including the word of length 0; the empty word ϵ. Two finite words $v = v_0...v_n$ and $v' = v'_0...v'_n$ can be *concatenated* just by putting them together: $vv' = v_0...v_nv'_0...v'_n$. We can also concatenate a finite word v with a right infinite word u: $vu = v_0...v_nu_0u_1....$ A finite word v is said to be a subword of another (finite or infinite) word u, if there exists two indexes $i < j$, such that $u_iu_{i+1}...u_j = v$. This is denoted by $v \sqsubseteq u$ (and $u \sqsupseteq v$). If the index i is equal to 0, we say that v is a *prefix* of u and is denoted by $v \sqsubset_p u$ (and $u \sqsupset_p v$). If the index j is equal to l, where l is the length of u, we say that v is a *suffix* of u and is denoted by $v \sqsubset_s u$ (and $u \sqsupset_s v$). It

is important to say that this definition of subword may vary in other works.

Now, let us introduce the *shift* function σ, which is defined both in $\Sigma^{\mathbb{Z}}$ and $\Sigma^{\mathbb{N}}$ by $\sigma(u)_i = u_{i+1}$. Given a subset $S \subseteq \Sigma^{\mathbb{Z}}$ (or $S \subseteq \Sigma^{\mathbb{N}}$), a formal language is defined by the subwords of sequences in S:

$$\mathcal{L}(S) = \{z \in \Sigma^* | (\exists w \in S) z \sqsubseteq w\}$$

Reciprocally, given a formal language L, a set of infinite sequences can be defined in $\Sigma^{\mathbb{M}}$ ($\mathbb{M} \in \{\mathbb{Z}, \mathbb{N}\}$):

$$S_L = \{w \in \Sigma^{\mathbb{M}} | (\forall z \sqsubseteq w) z \in L\}$$

When S satisfies $S_{\mathcal{L}(S)} = S$, it is called a *subshift*.

2.4 Topological dynamical system

A *topological dynamical system* (X, T, Ω) is a dynamical system (X, T) with a topology Ω such that T is continuous. When we refer to a topological dynamical system, we omit the collection Ω, which is defined by the open sets of the metric $d : X \times X \to \mathbb{N}$ in our cases, by the *balls* B_x as $\Omega = \{B_r(x) : x \in X, r > 0\}$.

Perfect set

A point $x \in S$, where $S \subset X$ is called an *isolated point* if there exists a neighborhood of x that does not contain any other point of S. Thus, a subset of a topological space is called a *perfect set* if it has no isolated points. In other words, given a perfect set S, any point can be approximated arbitrarily well by other points from the set.

2.5 Turing Machine

A Turing machine (TM) is a computational model that describes an abstract machine, which consists of a *head* that reads symbols from a *tape*, which are modified following the previously defined instructions of the machine. Turing machines are mainly used to define computability (can this problem be solved by a computer?). In this work, the dynamics of Turing machine are studied, therefore some preliminary considerations have to be made in order to specify Turing machines in the context of dynamical systems.

Formally, a Turing machine M is a tuple (Q, Σ, δ), where Q is a finite set of states, Σ is a finite set of symbols (a finite alphabet) and $\delta \subseteq Q \times \Sigma \times \Sigma \times Q \times \{-1, 0, +1\}$ is the transition relation of the machine.

Configuration of a Turing machine

A TM works over a tape, usually bi-infinite, full of symbols from Σ.

A *configuration* is an element $(r, i, w) \in Q \times \mathbb{Z} \times \Sigma^{\mathbb{Z}}$. A *finite configuration* is an element $(r, i, v) \in Q \times \{0, 1, ..., m-1\} \times \Sigma^m$ for some $m \in \mathbb{N}$. A *right semi-infinite configuration* is an element $(r, i, u) \in Q \times \mathbb{N} \times \Sigma^{\omega}$. A *left semi-infinite configuration* is an element $(r, i, u) \in Q \times (-\mathbb{N}) \times {}^{\omega}\Sigma$.

Instructions of a Turing machine

Basically, an *instruction* is what takes the machine from one configuration to another, in other words, instructions define the behavior of Turing machines, and particularly, the evolution of configurations.

Mathematically, an instruction is a quintuple $(r, s, s', r', c) \in Q \times \Sigma \times \Sigma \times Q \times \{-1, 0, +1\}$ and it can be applied to a configuration (r'', i, w) if $w_i = s$ and $r'' = r$, leading to the configuration $(r', i + c, w')$, where $w'_i = s'$ and $w'_k = w_k$ for all $k \neq i$. If the configuration is finite or (right or left) semi-finite with domain K, and $i + c \notin K$, then the instruction cannot be applied and the machine halts. If no instruction can be applied, the machine halts.

Definition 1. *For configurations x, y, we say that a Turing machine M reaches configuration y from x if y is the result of evolving M on x over a finite number of steps, and we denote this by $x \vdash^* y$. This notation is considered for finite, semi-finite and infinite configurations.*

In the computational context, Turing machines have an initial and a final state, but since we are studying its dynamics, we omit these definitions.

Deterministic Turing machine

A Turing machine M is *deterministic* if for any configuration (r, i, w), at most one instruction can be applied. This is equivalent to give δ as a (possibly partial) function $\delta : Q \times \Sigma \to Q \times \Sigma \times \{-1, 0, +1\}$.

Complete Turing machine

A Turing machine M is *complete* if for each configuration (r, i, w), at least one instruction can be applied. This is equivalent saying that the machine never halts.

Injective Turing machine

A Turing machine M is *injective* when δ is injective. This is:

$$(\forall q, q' \in Q)(\forall s, s' \in \Sigma) : \delta(q, s) = \delta(q', s') \Rightarrow q = q' \wedge s = s'$$

This is equivalent to say that every configuration comes from at most one pre-image or previous configuration.

Reversible Turing machine
A Turing Machine M is *reversible* if it is deterministic and injective.

In this work, we will use the definition of reverse Turing machine from [3]. A reversible Turing machine can be characterized by a pair (ρ, μ), where $\rho : Q \times \Sigma \to Q \times \Sigma$ is a partial injective function and $\mu : Q \to \{-1, 0, +1\}$ is a partial function, such that δ is characterized by all the instructions of the form $(r, s, s', r', \mu(r'))$ where $r \in Q$, $s \in \Sigma$ and $(r', s') = \rho(r, s)$.

Indeed, the movement portion of the instructions depends on the state at which it goes in a reversible Turing machine; if not, the Turing machine has a configuration with more than one pre-image, and therefore it would be not injective. Now we can define the following:

Definition 2. *The reverse of a Turing machine $M = (Q, \Sigma, \delta)$ is defined by $M^{-1} = (Q, \Sigma, \delta^{-1})$, where $(r', s', s, r, -\mu(r)) \in \delta^{-1}$ if and only if $r \in Q$, $s \in \Sigma$ and $(r', s') = \rho(r, s)$.*

The reverse machine is called this way because it reverses the computation. In this case, we need to define the function $\phi : (\Sigma^{\mathbb{Z}}, \mathbb{Z}, Q) \to (\Sigma^{\mathbb{Z}}, \mathbb{Z}, Q)$ as $\phi(w, i, r) = (w, i - \mu(r), r)$, then the reverse computation is obtained by applying $\phi^{-1} \circ M^{-1} \circ \phi$.

2.6 Turing Machine as a Dynamical System

Topological dynamical systems of Turing machine is a paradigm formalized by [8] (but firstly introduced by [9]) and it gives us a strong tool for the study of the dynamics of Turing machine. In this research, we will consider the dynamical system called *Turing machine with Moving Tape* (TMT), which consists in putting the head at the center of the tape (the 0 position) and only moving the tape instead.

The dynamical system (X, T) for TMT consists in: $X \subseteq {}^{\omega}\Sigma \times Q \times \Sigma^{\omega}$ and $T : X \to X$ is the application of δ by moving the tape instead of the head. An element from ${}^{\omega}\Sigma \times Q \times \Sigma^{\omega}$ is called a *TMT configuration*.

To have a better understanding of how this system works, let us specify the way that instructions are applied: instruction (r, u_0, s', r', c)

is applied to a TMT configuration $(...w_2w_1w_0, r, u_0u_1u_2...)$ resulting in:

- if $c = -1$, $(...w_3w_2w_1, r', w_0s'u_1...)$
- if $c = 0$, $(...w_2w_1w_0, r', s'u_1u_2...)$
- if $c = +1$, $(...w_1w_0s', r', u_1u_2u_3...)$

We call a *finite configuration* of TMT, a tuple $(v, r, v') \in {}^*\Sigma \times Q \times \Sigma^*$. Additionally, the metric used to measure the distance between a pair of points $(w, r, u), (w', r', u') \in X$ is, considering $i = min\{j : w_{-j} \neq w'_{-j} \vee u_j \neq u'_j, j \in \mathbb{N}\}$:

$$d((w, r, u), (w', r', u')) = \begin{cases} 1 & \text{if } r \neq r' \\ 2^{-i} & \text{in other cases} \end{cases}$$

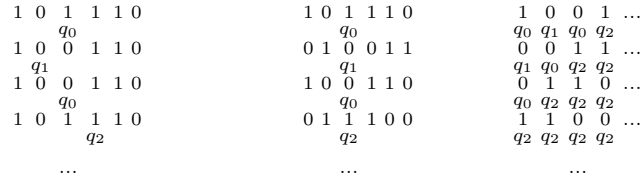
As we can see, this is a modification of the Cantor metric.

2.7 The *t*-shift

Taking into account the TMT dynamical system, we can define the projection $\pi : X \to Q \times \Sigma$ by $\pi(w, r, w') = (r, w'_0)$.

The *t-shift*, denoted by $S_T \subseteq (Q \times \Sigma)^{\mathbb{N}}$, is the sets of orbits $\tau(x) = (\pi(T^n(x)))_{n \in \mathbb{N}}$ for $x \in X$. In other terms, $S_T = \{\tau(x) : x \in X\}$.

In figure 1, a comparison among the classic Turing machine model, TMT and t-shift can be seen through an example.



Turing machine model TMT model t-shift

FIGURE 1: Examples of the evolution of a TM in its dynamical models (the classic one, TMT and t-shift). The represented machine has the instructions $(q_0, 0, 1, q_2, +1)$, $(q_0, 1, 0, q_1, -1)$, $(q_1, 0, 0, q_0, +1)$ and $(q_2, \alpha, \alpha, q_2, +1)$, $\forall \alpha \in \Sigma$.

As we can see, the t-shift stores the read symbol and the current inner state at each step of computation for every possible configuration i.e. it contains relevant information of the computation history of the machine.

Cylinder

In this context, a ball is called a *cylinder* and it is defined by $B_r(x) = \{y \in {}^{\omega}\Sigma \times Q \times \Sigma^{\omega} : d(x,y) < r\}$, but we can also define it in a more intuitive way: Given a finite configuration $(v, r, v') \in \Sigma^* \times Q \times \Sigma^*$, its cylinder is:

$$[v, r, v'] = \{(w, r', w') \in {}^{\omega}\Sigma \times Q \times \Sigma^{\omega} : (r = r')(w \sqsupset_s v)(w' \sqsupset_p v')\}$$

2.8 Turing machine dynamical and topological properties

Here, we present the properties linked with the Turing machine dynamical system that are the center of our study.

Aperiodicity

The first of this properties is *aperiodicity* which says that a dynamical system does not have any periodic point. In the Turing machine context, we will call *periodic configuration* instead of periodic point, then a Turing machine is aperiodic if it does not have any periodic configuration. In mathematical terms:

$$(\forall x \in X)(\forall n > 0) : T^n(x) \neq x$$

Topological transitivity

A dynamical system (X, T) is topologically transitive if there exists a point $x \in X$ such that for all point $y \in X$ we got that $y \in \overline{\mathcal{O}(x)}$. When this happens, we say that x is a *transitive point* and $\mathcal{O}(x)$ is *dense*.

Now, let us contextualize this property for the TMT dynamical system:

Definition 3. *Let (X, T) be a TMT dynamical system. (X, T) is topologically transitive if*

$$(\forall u, v \in \Sigma^* \times Q \times \Sigma^*)(\exists x \in [u])(\exists n > 0) : T^n(x) \in [v]$$

or, equivalently:

$$(\exists x \in X)(\forall u \in \Sigma^* \times Q \times \Sigma^*)(\exists n > 0) : T^n(x) \in [u]$$

The previous definitions are indeed equivalent, due that we work over a perfect set [1].

Topological minimality

If every point of a dynamical system is topologically transitive, then it is topologically minimal.

Definition 4. *Let (X, T) be a TMT dynamical system. (X, T) is topologically minimal if*

$$(\forall x \in X)(\forall u \in \Sigma^* \times Q \times \Sigma^*)(\exists n > 0) : T^n(x) \in [u]$$

Time-symmetry

Time-symmetry is a property firstly studied in physical systems and it is considered stronger than reversibility. When a system presents this property, it is indistinguishable if the system goes forward or backward in time. The definition for Turing machines has been taken from [3].

Definition 5. *A reversible Turing machine $M = (Q, \Sigma, \delta)$ is said to be time-symmetric if there exists involutions $h_Q : Q \to Q$ and $h_\Sigma : \Sigma \to \Sigma$ such that:*

$$(h_Q(r), h_\Sigma(s), h_\Sigma(s'), h_Q(r'), c) \in \delta^{-1} \Leftrightarrow (r, s, s', r', c) \in \delta$$

Substitutive subshift

A *substitution* is a morphism $\phi : \Sigma^* \to \Sigma^*$, which can be extended to $\Sigma^{\mathbb{N}}$. A fixed point of ϕ is a word $w \in \Sigma^{\mathbb{N}}$ such that $\phi(w) = w$. A subshift is substitutive if it is the closure of the orbit of a fixed point of some substitution. In that case, we can define the subshift with that substitution.

3 BINSMART TOPOLOGICAL AND DYNAMICAL PROPERTIES

3.1 The BinSmart machine

In this section, we introduce the Binary Smart (BinSmart) machine which is the main object of study in this work. This Turing machine is based on another machine known as SMART machine [3], but it is not a simple recoding, as it would imply a loss in reversibility or completeness.

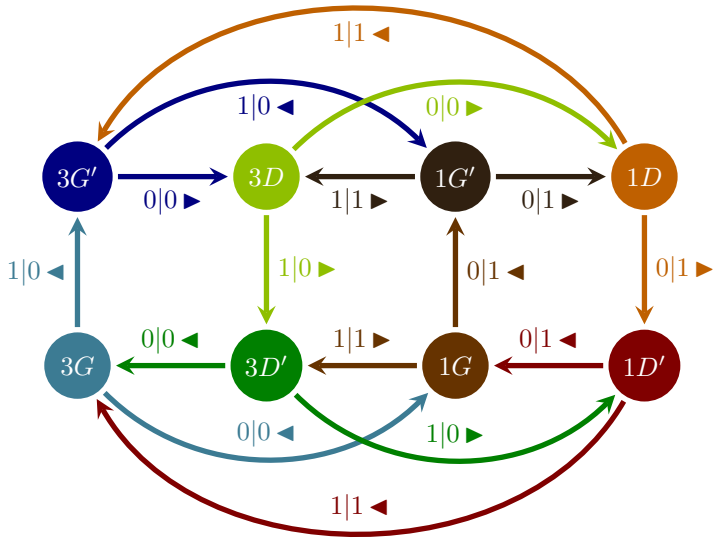


FIGURE 2: Binary Smart(BinSmart): We gave it this name because it has a similar behavior than SMART machine, but with just two symbols. An arrow from r to r' labelled $\alpha|\alpha' c$ represents the instruction $(r, \alpha, \alpha', r', c)$ of the machine.

We remark the symmetry between states $1D$ and $1G$, between $1D'$ and $1G'$, between $3D$ and $3G$ and between $3D'$ and $3G'$. For example, the states $1D$ and $1G$ read and write exactly the same symbols but have opposite moving directions; this can be extended to the rest of the states.

In this section, we try to follow the same steps as [3] in order to proof the desired properties, but, as we conclude that Time-Symmetry is not present in BinSmart, we change our approach to decide about Topological Minimality.

3.2 BinSmart's behavior

The behavior of the BinSmart machine consists of applying bounded searches of 1s. To describe this behavior, considering $s \in \{0, 1\}$, the following propositions are defined:

$$D_1(n): \begin{pmatrix} 0 & & 0^n & 1 \\ 1D & & & \end{pmatrix} \vdash^* \begin{pmatrix} 1 & & 0^n & 1 \\ 3G & & & \end{pmatrix} \qquad G_1(n): \begin{pmatrix} 1 & 0^n & & 0 \\ & & & 1G \end{pmatrix} \vdash^* \begin{pmatrix} 1 & 0^n & & 1 \\ & & & 3D \end{pmatrix}$$

$$D_1'(n): \begin{pmatrix} 1 & & 0 & & 0^n & s \\ & & 1D' & & & \end{pmatrix} \vdash^* \begin{pmatrix} 1 & 0 & 0^n & & s \\ & & & & 1D' \end{pmatrix} \qquad G_1'(n): \begin{pmatrix} s & 0^n & & 0 & & 1 \\ & & & 1G' & & \end{pmatrix} \vdash^* \begin{pmatrix} & s & 0^n & 0 & 1 \\ & 1G' & & & \end{pmatrix}$$

$$D_3(n): \begin{pmatrix} 0 & & 0^n & 1 \\ 3D & & & \end{pmatrix} \vdash^* \begin{pmatrix} 0 & 0^n & & 1 \\ & & & 3D \end{pmatrix} \qquad G_3(n): \begin{pmatrix} 1 & 0^n & & 0 \\ & & & 3G \end{pmatrix} \vdash^* \begin{pmatrix} 1 & & 0^n & 0 \\ 3G & & & \end{pmatrix}$$

$$D_3'(n): \begin{pmatrix} 1 & 0^n & & 0 \\ & & & 3D' \end{pmatrix} \vdash^* \begin{pmatrix} 1 & & 0^n & 0 \\ 3G & & & \end{pmatrix} \qquad G_3'(n): \begin{pmatrix} 0 & & 0^n & 1 \\ 3G' & & & \end{pmatrix} \vdash^* \begin{pmatrix} 0 & 0^n & & 1 \\ & & & 3D \end{pmatrix}$$

Lemma 1. *$D_1(n)$, $G_1(n)$, $D_1'(n)$, $G_1'(n)$, $D_3(n)$, $G_3(n)$, $D_3'(n)$ and $G_3'(n)$ are true for all $n \in \mathbb{N}$.*

Proof. Since $D_1(n)$, $D_1'(n)$, $D_3(n)$ and $D_3'(n)$ are symmetrical to $G_1(n)$, $G_1'(n)$, $G_3(n)$ and $G_3'(n)$, we will do the proofs just for the first ones. We prove $D_1'(n)$ and $D_3(n)$ by making an induction over n. The basis can be done by hand by simulating the machine. Let us suppose that these propositions are true for $n-1$ and for $n-2$. First we prove $D_1'(n)$.

$$\begin{pmatrix} 1 & & 0 & & 0^{n-3} & 0 & 0 & 0 & s \\ & & 1D' & & & & & & \end{pmatrix}$$
$$\vdash^{D_1'(n-1)}$$
$$\begin{pmatrix} 1 & 0 & 0^{n-3} & 0 & 0 & & 0 & & s \\ & & & & & & 1D' & & \end{pmatrix}$$
$$\vdash^{2}$$
$$\begin{pmatrix} 1 & 0 & 0^{n-3} & & 0 & & 1 & 1 & s \\ & & & & 1G' & & & & \end{pmatrix}$$
$$\vdash^{G_1'(n-2)}$$
$$\begin{pmatrix} & 1 & & 0 & 0^{n-3} & 0 & 1 & 1 & s \\ & 1G' & & & & & & & \end{pmatrix}$$
$$\vdash$$
$$\begin{pmatrix} 1 & & 0 & & 0^{n-3} & 0 & 1 & 1 & s \\ & 3D & & & & & & & \end{pmatrix}$$
$$\vdash^{D_3(n-2)}$$
$$\begin{pmatrix} 1 & 0 & 0^{n-3} & 0 & & 1 & & 1 & s \\ & & & & 3D & & & & \end{pmatrix}$$
$$\vdash$$
$$\begin{pmatrix} 1 & 0 & 0^{n-3} & 0 & 0 & 0 & & s \\ & & & & & & 1D' & \end{pmatrix}$$

Now, for $D_3(n)$

$$\begin{pmatrix} & 0 & & 0 & 0 & 0^{n-3} & 0 & 1 \\ & 3D & & & & & & \end{pmatrix}$$
$$\vdash^{2}$$
$$\begin{pmatrix} 0 & 1 & & 0 & & 0^{n-3} & 0 & 1 \\ & & & 1D' & & & & \end{pmatrix}$$
$$\vdash^{D_1'(n-2)}$$
$$\begin{pmatrix} 0 & 1 & 0 & 0^{n-3} & 0 & & 1 \\ & & & & & & 1D' \end{pmatrix}$$
$$\vdash$$
$$\begin{pmatrix} 0 & 1 & 0 & 0^{n-3} & & 0 & & 1 \\ & & & & & 3G & & \end{pmatrix}$$
$$\vdash^{G_3(n-2)}$$
$$\begin{pmatrix} 0 & & 1 & & 0 & 0^{n-3} & 0 & 1 \\ & & 3G & & & & & \end{pmatrix}$$
$$\vdash^{2}$$
$$\begin{pmatrix} 0 & & 0 & & 0 & 0^{n-3} & 0 & 1 \\ & & 3D & & & & & \end{pmatrix}$$
$$\vdash^{D_3(n-1)}$$
$$\begin{pmatrix} 0 & 0 & 0 & 0^{n-3} & 0 & & 1 \\ & & & & & & 3D \end{pmatrix}$$

Since $D_1(n)$ and $D'_3(n)$ are not recursive, we prove them directly. Let us prove $D_1(n)$.

$$\begin{array}{ll}
\left(\begin{smallmatrix} 0 \\ 1D \end{smallmatrix}\ 0\ 0^{n-2}\ 0\ 1\right) & \\
 & \text{One Step} \\
\left(1\ \begin{smallmatrix} 0 \\ 1D' \end{smallmatrix}\ 0^{n-2}\ 0\ 1\right) & \\
 & \text{Apply } D'_1(n-1) \\
\left(1\ 0\ 0^{n-2}\ 0\ \begin{smallmatrix} 1 \\ 1D' \end{smallmatrix}\right) & \\
 & \text{One step} \\
\left(1\ 0\ 0^{n-2}\ \begin{smallmatrix} 0 \\ 3G \end{smallmatrix}\ 1\right) & \\
 & \text{Apply } G_3(n-1) \\
\left(\begin{smallmatrix} 1 \\ 3G \end{smallmatrix}\ 0\ 0^{n-2}\ 0\ 1\right) &
\end{array}$$

Now, for $D'_3(n)$

$$\begin{array}{ll}
\left(1\ 0^{n-1}\ 0\ \begin{smallmatrix} 0 \\ 3D' \end{smallmatrix}\right) & \\
 & \text{One Step} \\
\left(1\ 0^{n-1}\ \begin{smallmatrix} 0 \\ 3G \end{smallmatrix}\ 1\right) & \\
 & \text{Apply } G_3(n-1) \\
\left(\begin{smallmatrix} 1 \\ 3G \end{smallmatrix}\ 0^{n-1}\ 0\ 0\right) &
\end{array}$$

□

3.3 Aperiodicity

Before proving that BinSmart does not have any periodic configuration, we prove aperiodicity in two particular but important points.

Lemma 2. $\left(1\ 0^n\ \begin{smallmatrix} 1 \\ 3D' \end{smallmatrix}\ 0\right) \vdash^* \left(1\ 0^n\ 0\ \begin{smallmatrix} 1 \\ 3D' \end{smallmatrix}\right)$

Proof.

$$\begin{array}{ll}
\left(1\ 0^n\ \begin{smallmatrix} 1 \\ 3D' \end{smallmatrix}\ 0\right) & \\
 & \vdash^2 \\
\left(1\ 0^n\ \begin{smallmatrix} 0 \\ 1G \end{smallmatrix}\ 1\right) & \\
 & \vdash^{G_1(n)} \\
\left(1\ 0^n\ \begin{smallmatrix} 1 \\ 3D \end{smallmatrix}\ 1\right) & \\
 & \vdash \\
\left(1\ 0^n\ 0\ \begin{smallmatrix} 1 \\ 3D' \end{smallmatrix}\right) &
\end{array}$$

□

Lemma 3. *The semi-infinite configuration* $\left(\begin{smallmatrix} 0 \\ 1D \end{smallmatrix}\ 0^\omega\right)$ *is not periodic.*

Proof. Starting with this configuration, the machine will evolve into $\left(1\ 0\ \begin{smallmatrix} 1 \\ 3D' \end{smallmatrix}\ 0^\omega\right)$ after 8 steps. Now we can apply Lemma 2 and see that the evolution of this configuration is in fact not periodic.

□

In order to generalize aperiodicity to any configuration, we will prove that arbitrary large blocks of 0s appear regardless of the context and in a recurrent way.

Lemma 4. *If we define, for every $n \geq 0$, the set $C_n = \{x | x \in \left(\begin{smallmatrix} 0 & 0^n \\ 1D & \end{smallmatrix}\right) \cup \left(\begin{smallmatrix} 0^n & 0 \\ & 1G \end{smallmatrix}\right)\}$, then for every $x \in C_n$, either x or its orbit will eventually visit C_m for arbitrary large m.*

Proof. Since the states $1D$ and $1G$ are symmetrical, we just make the proof for the first one. We use $s_0, s_1, s_2, s_3 \in \{0, 1\}$ as variables.

$$\left(\begin{smallmatrix} s_0 & s_1 & 0 & 0^n & 1 & s_2 & s_3 \\ & & 1D & & & & \end{smallmatrix}\right) \vdash^{D_1(n)} \left(\begin{smallmatrix} s_0 & s_1 & 1 & 0^n & 1 & s_2 & s_3 \\ & & 3G & & & & \end{smallmatrix}\right) \vdash \left(\begin{smallmatrix} s_0 & s_1 & 0 & 0 & 0^n & 1 & s_2 & s_3 \\ & & 3G' & & & & & \end{smallmatrix}\right)$$

if $s_1 = 0$

$$\left(\begin{smallmatrix} s_0 & 0 & 0 & 0 & 0^n & 1 & s_2 & s_3 \\ & & 3G' & & & & & \end{smallmatrix}\right) \vdash^{G'_3(n+1)} \left(\begin{smallmatrix} s_0 & 0 & 0 & 0 & 0^n & 1 & s_2 & s_3 \\ & & & & & 3D & & \end{smallmatrix}\right) \vdash \left(\begin{smallmatrix} s_0 & 0 & 0 & 0 & 0^n & 0 & s_2 & s_3 \\ & & & & & & 3D' & \end{smallmatrix}\right)$$

if $s_2 = 1$

$$\left(\begin{smallmatrix} s_0 & 0 & 0 & 0 & 0^n & 0 & 1 & s_3 \\ & & & & & & 3D' & \end{smallmatrix}\right) \vdash \left(\begin{smallmatrix} s_0 & 0 & 0 & 0 & 0^n & 0 & 0 & s_3 \\ & & & & & & & 1D' \end{smallmatrix}\right)$$

if $s_3 = 0$

$$\left(\begin{smallmatrix} s_0 & 0 & 0 & 0 & 0^n & 0 & 0 & 0 \\ & & & & & & & 1D' \end{smallmatrix}\right) \vdash \left(\begin{smallmatrix} s_0 & 0 & 0 & 0 & 0^n & 0 & 0 & 1 \\ & & & & & & 1G & \end{smallmatrix}\right) \blacksquare$$

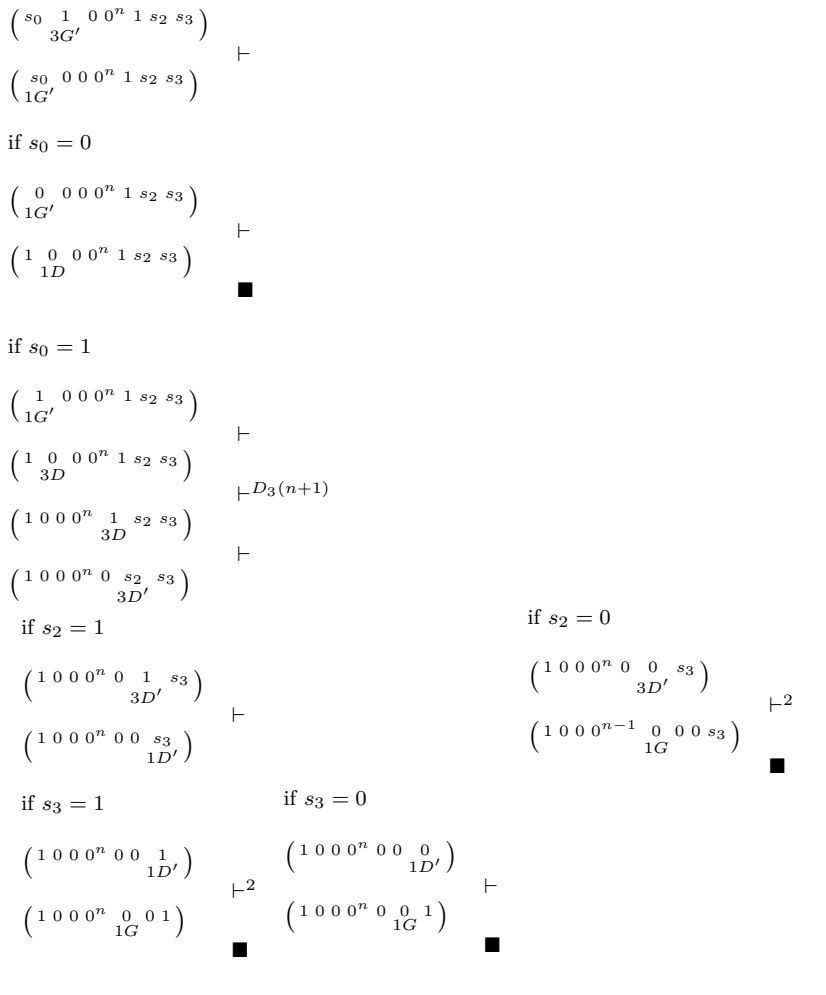
if $s_3 = 1$

$$\left(\begin{smallmatrix} s_0 & 0 & 0 & 0 & 0^n & 0 & 0 & 1 \\ & & & & & & & 1D' \end{smallmatrix}\right) \vdash^2 \left(\begin{smallmatrix} s_0 & 0 & 0 & 0 & 0^n & 0 & 0 & 1 \\ & & & & & & 1G & \end{smallmatrix}\right) \blacksquare$$

if $s_2 = 0$

$$\left(\begin{smallmatrix} s_0 & 0 & 0 & 0 & 0^n & 0 & 0 & s_3 \\ & & & & & & 3D' & \end{smallmatrix}\right) \vdash^2 \left(\begin{smallmatrix} s_0 & 0 & 0 & 0 & 0^{n-1} & 0 & 0 & 0 & s_3 \\ & & & & & & 1G & & \end{smallmatrix}\right) \blacksquare$$

Now we study the case $s_1 = 1$



Theorem 1. *The BinSmart machine has no periodic points.*

Proof. Consider an arbitrary configuration, after at most 11 steps, the machine will be reading a 0 symbol in either state $1D$ or $1G$, in other words, it arrives to one of the sets C_n defined in Lemma 4. The amount of 0s will grow then, expanding to the left or to the right. At some point the machine will either reach a configuration of the form $\left(\begin{smallmatrix} 0 \\ 1D \end{smallmatrix}\ 0^{\omega}\right)$ (or its symmetric), which we know to be aperiodic from Lemma 3, or it will pass by an infinite sequence of configurations of the form $\left(\begin{smallmatrix} 0 \\ 1D \end{smallmatrix}\ 0^{n}\right)$

, with $n \geq 0$, (or its symmetric), implying that its behavior is not periodic. □

3.4 Time-Symmetry

Let us present the reverse Turing machine of BinSmart in figure 3, in order to compare it with possible involutions of the original machine.

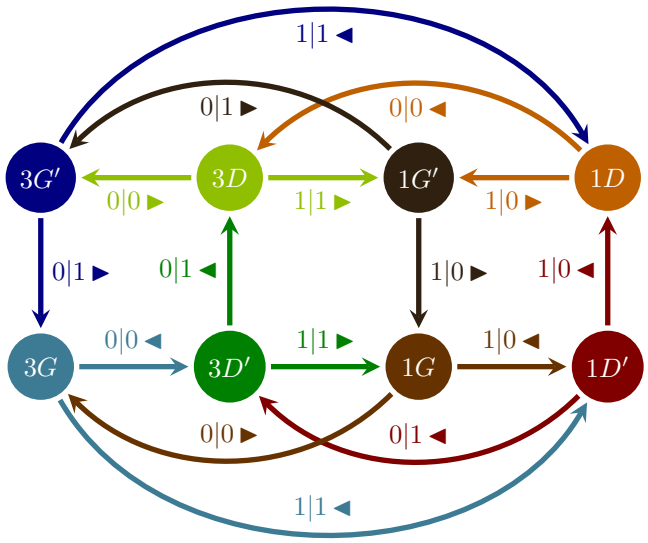


FIGURE 3: Reverse Binary Smart.

Proposition 1. *The BinSmart machine is not time-symmetric.*

Proof. We know that the instructions $(1D', 0, 1, 1G, \blacktriangleleft)$ and $(1D', 1, 1, 3G, \blacktriangleleft)$ are in δ, then, if the BinSmart machine is time-symmetric, there must exist two involutions h_Q and h_Σ such that $(h_Q(1D'), h_\Sigma(0), h_\Sigma(1), h_Q(1G), \blacktriangleleft)$ and $(h_Q(1D'), h_\Sigma(1), h_\Sigma(1), h_Q(3G), \blacktriangleleft)$ are in δ^{-1}. But there does not exist an involution h_Q that satisfies this condition neither with the involution $h_\Sigma : \{0 \to 0\}$ or with the involution $h_\Sigma : \{0 \to 1\}$. □

3.5 Topological transitivity and minimality

Now, as we proved that BinSmart is not Time-Symmetric, we cannot follow the same argument used in [3].

In this fashion, we directly prove topological minimality which implies topological transitivity. To prove BinSmart minimality, we need to prove that every bi-infinite configuration reaches any finite configuration. To do that, we demonstrate that an arbitrary bi-infinite configuration x and an arbitrary finite configuration u reach another but 'identical' configuration v in different times. We also prove that u evolves into v faster than x does. Since the machine is reversible, there is only one path to reach a specific configuration, then if x and u evolve into v, and u do it faster than x, implies that x reaches u. Since x and u are arbitrary, this argument is enough to prove minimality. To demonstrate this, the following lemmas are stated.

Lemma 5. *Every $x \in X$ will reach $\left(\begin{smallmatrix} 0 \\ 1D \end{smallmatrix}\ 0^p\right)$, for all $p \in \mathbb{N}$.*

Proof. As we know from Theorem 1, after at most 11 steps, any configuration arrives in one of the sets C_n for some n, then we apply Lemma 4 in order to reach C_m for any m, getting one of the following configurations (disregarding the context):

(i) $\left(\begin{smallmatrix} 0 \\ 1D \end{smallmatrix}\ 0^m\right)$

(ii) $\left(0^m\ \begin{smallmatrix} 0 \\ 1G \end{smallmatrix}\right)$

If we reach (i), we are done, so let us see the case (ii):

$$\begin{aligned}
&\left(0^m\ \begin{smallmatrix} 0 \\ 1G \end{smallmatrix}\right) \\
&\qquad \vdash^3 \\
&\left(0^{m-1}\ \begin{smallmatrix} 1 \\ 3G' \end{smallmatrix}\ 1\right) \\
&\qquad \text{Apply Lemma 2 } (m-2) \text{ times} \\
&\left(0\ \begin{smallmatrix} 1 \\ 3G' \end{smallmatrix}\ 0^{m-2}\ 1\right) \\
&\qquad \vdash^2 \\
&\left(1\ \begin{smallmatrix} 0 \\ 1D \end{smallmatrix}\ 0^{m-2}\ 1\right)
\end{aligned}$$

■

In this case we got that $p = m - 2$. Since we can reach this configuration for any m, we can do it for any p.

□

Lemma 6. $\left(\begin{smallmatrix} 0 \\ 1D \end{smallmatrix}\ 0^n\right) \vdash^* \left(1\ 0^m\ \begin{smallmatrix} 1 \\ 3D' \end{smallmatrix}\ 0^{n-m-1}\right)$, *for any $m < n$*

Proof. Starting from $\left(\begin{smallmatrix} 0 \\ 1D \end{smallmatrix}\ 0^n\right)$, after 3 steps we reach $\left(1\ \begin{smallmatrix} 1 \\ 3D' \end{smallmatrix}\ 0^{n-1}\right)$, then we apply Lemma 2 m times, in an iterative way, to obtain: $\left(1\ 0^m\ \begin{smallmatrix} 1 \\ 3D' \end{smallmatrix}\ 0^{n-m-1}\right)$.

□

Lemma 7. *Considering an arbitrary finite word $w \in \{0,1\}^*$ of length l, the following statements are true:*

(i) $\left(\begin{smallmatrix} 1\,1 & 0 & 0^n\ w_1\ \dots\ w_l\ 1 \\ & 1D & \end{smallmatrix}\right) \vdash^* \left(\begin{smallmatrix} 1\,0\,0\,0^n\ 0 & w_2\ \dots\ w_l\ 1 \\ & 3D' & \end{smallmatrix}\right)$

(ii) $\left(\begin{smallmatrix} 1\ w_1\ \dots\ w_l & 0 & 0^n\ 1 \\ & 1D & \end{smallmatrix}\right) \vdash^* \left(\begin{smallmatrix} 1\ w_1\ \dots & w_l & 0\ 0^n\ 1 \\ & 3G' & \end{smallmatrix}\right)$

(iii) $\left(\begin{smallmatrix} 1\ w_1\ \dots\ w_l\ 0^n & 0 & 1\ 1 \\ & 1G & \end{smallmatrix}\right) \vdash^* \left(\begin{smallmatrix} 1\ w_1\ \dots\ w_{l-1}\ 0\ 0^n\ 0\ 0\ 1 \\ & 3G' & \end{smallmatrix}\right)$

(iv) $\left(\begin{smallmatrix} 1\ 0^n & 0 & w_1\ \dots\ w_l\ 1 \\ & 1G & \end{smallmatrix}\right) \vdash^* \left(\begin{smallmatrix} 1\ 0^n\ 0 & w_1 & \dots\ w_l\ 1 \\ & 3D' & \end{smallmatrix}\right)$

Proof. Since (i) and (ii) are symmetrical to (iii) and (iv) we will do the proofs only for the first two.
Case (i):

$$\left(\begin{smallmatrix} 1\,1 & 0 & 0^n\ w_1\ \dots\ w_l\ 1 \\ & 1D & \end{smallmatrix}\right)$$
$$\vdash^{D_1(n)} (w_1 \text{ have to be equal to } 1)$$
$$\left(\begin{smallmatrix} 1\,1 & 1 & 0^n\ 1\ w_2\ \dots\ w_l\ 1 \\ & 3G & \end{smallmatrix}\right)$$
$$\vdash^{3}$$
$$\left(\begin{smallmatrix} 1 & 0 & 0\ 0^n\ 1\ w_2\ \dots\ w_l\ 1 \\ & 3D & \end{smallmatrix}\right)$$
$$\vdash^{D_3(n+1)}$$
$$\left(\begin{smallmatrix} 1\,0\,0\,0^n & 1 & w_2\ \dots\ w_l\ 1 \\ & 3D & \end{smallmatrix}\right)$$
$$\vdash$$
$$\left(\begin{smallmatrix} 1\,0\,0\,0^n\ 0 & w_2 & \dots\ w_l\ 1 \\ & 3D' & \end{smallmatrix}\right)$$
$$\blacksquare$$

Case (ii):

$$\left(\begin{smallmatrix} 1\ w_1\ \dots\ w_l & 0 & 0^n\ 1 \\ & 1D & \end{smallmatrix}\right)$$
$$\vdash^{D_1(n)}$$
$$\left(\begin{smallmatrix} 1\ w_1\ \dots\ w_l & 1 & 0^n\ 1 \\ & 3G & \end{smallmatrix}\right)$$
$$\vdash$$
$$\left(\begin{smallmatrix} 1\ w_1\ \dots & w_l & 0\ 0^n\ 1 \\ & 3G' & \end{smallmatrix}\right)$$
$$\blacksquare$$

□

Lemma 8. *Considering an arbitrary finite word $w \in \{0,1\}^*$ of length l, we got that:* $\left(\begin{smallmatrix} 1\,1\,1\,0^n & w_1 & \dots\ w_l \\ & 3D' & \end{smallmatrix}\right) \vdash^* \left(\begin{smallmatrix} 1\,0\,0\,0^{n+c+2} & w_{c+3} & \dots\ w_l \\ & 3D' & \end{smallmatrix}\right)$, *where c is the amount of 0 symbols between w_1 and the first 1 symbol to the right.*

Proof. First, let us see the case $w_1 = 0$

$$\begin{array}{ll}
\left(\begin{smallmatrix} 1\,1\,1\,0^n & 0 & w_2\,\ldots\,w_l \\ & 3D' & \end{smallmatrix}\right) & \\
 & \vdash^{D'_3(n)} \\
\left(\begin{smallmatrix} 1\,1 & 1 & 0^n\,0\,w_2\,\ldots\,w_l \\ & 3G & \end{smallmatrix}\right) & \qquad (*) \\
 & \vdash^{3} \\
\left(\begin{smallmatrix} 1 & 0 & 0\,0^n\,0\,w_2\,\ldots\,w_l \\ & 3D & \end{smallmatrix}\right) & \\
 & \vdash^{D_3(n+c+2)} \\
\left(\begin{smallmatrix} 1\,0\,0\,0^n\,0\,0^c & 1 & w_{c+3}\,\ldots\,w_l \\ & 3D & \end{smallmatrix}\right) & \\
 & \vdash \\
\left(\begin{smallmatrix} 1\,0\,0\,0^n\,0\,0^c\,0 & w_{c+3}\,\ldots\,w_l \\ & 3D' & \end{smallmatrix}\right) & \\
 & \blacksquare
\end{array}$$

Now, let us see the case $w_1 = 1$

$$\begin{array}{ll}
\left(\begin{smallmatrix} 1\,1\,1\,0^n & 1 & w_2\,\ldots\,w_l \\ & 3D' & \end{smallmatrix}\right) & \\
 & \vdash \\
\left(\begin{smallmatrix} 1\,1\,1\,0^n\,0 & w_2\,\ldots\,w_l \\ & 1D' & \end{smallmatrix}\right) &
\end{array}$$

If $w_2 = 1$

$$\begin{array}{ll}
\left(\begin{smallmatrix} 1\,1\,1\,0^n\,0 & 1 & w_3\,\ldots\,w_l \\ & 1D' & \end{smallmatrix}\right) & \\
 & \vdash \\
\left(\begin{smallmatrix} 1\,1\,1\,0^n & 0 & 1\,w_3\,\ldots\,w_l \\ & 3G & \end{smallmatrix}\right) & \\
 & \vdash^{G_3(n)} \\
\left(\begin{smallmatrix} 1\,1 & 1 & 0^n\,0\,1\,w_3\,\ldots\,w_l \\ & 3G & \end{smallmatrix}\right) & \\
 & \text{Which reduces to } (*)
\end{array}$$

If $w_2 = 0$, we can use Lemma 2 c times, obtaining $\left(\begin{smallmatrix} 1\,1\,1\,0^{n+c} & 1 & 1\,w_{c+3}\,\ldots\,w_l \\ & 3D' & \end{smallmatrix}\right)$, which reduces to the case $w_1 = w_2 = 1$. □

Corollary 1. $\left(\begin{smallmatrix} 1^k\,1\,0^n & w_1 & \ldots\,w_l\,1 \\ & 3D' & \end{smallmatrix}\right) \vdash^* \left(\begin{smallmatrix} 1\,0^{k+n+l} & 1 \\ & 3D' \end{smallmatrix}\right)$, *with* $k = 2 \cdot |w|_1$.

Lemma 9. *Considering an arbitrary finite word* $w \in \{0,1\}^*$ *of length* m*, for every* $i \in \{1,2,3,...,m\}$ *and every* $r \in Q$*, there exists* $k_1, k_2 \in \mathbb{N}$ *such that, every configuration of the form* $x = \left(\begin{smallmatrix} 1^{k_1+1}\,w_1\,\ldots & \underset{r}{w_i} & \ldots\,w_m\,1^{k_2+1} \end{smallmatrix}\right)$ *evolves into* $\left(\begin{smallmatrix} 1\,0^{k_1+k_2+m} & 1 \\ & 3D' \end{smallmatrix}\right)$ *or into* $\left(\begin{smallmatrix} 1 & 0^{k_1+k_2+m}\,1 \\ 3G' & \end{smallmatrix}\right)$.

Proof. As we know from Theorem 1, every configuration evolves into a configuration that belongs to one of the sets C_n defined in Lemma 4, then the number of 0's will increase either to the right or to the left. We will call $w \in \{0,1\}^*$ to the part of the symbols that have not been turned into 0's, and l to the length of w. Then, we will reach one of the following configurations:

(i) $\left(\begin{smallmatrix} 1^{k_1+1} & \underset{1D}{0} & 0^n & w_1 & \dots & w_l & 1^{k_2+1} \end{smallmatrix}\right)$

(ii) $\left(\begin{smallmatrix} 1^{k_1+1} & w_1 & \dots & w_l & \underset{1D}{0} & 0^n & 1^{k_2+1} \end{smallmatrix}\right)$

(iii) $\left(\begin{smallmatrix} 1^{k_1+1} & w_1 & \dots & w_l & 0^n & \underset{1G}{0} & 1^{k_2+1} \end{smallmatrix}\right)$

(iv) $\left(\begin{smallmatrix} 1^{k_1+1} & 0^n & \underset{1G}{0} & w_1 & \dots & w_l & 1^{k_2+1} \end{smallmatrix}\right)$

where $n+l+1=m$. As before, we will do the proof only for the cases (i) and (ii) since they are symmetric to (iii) and (iv). At this point, we can apply Lemma 7 obtaining the following configurations:

(a) $\left(\begin{smallmatrix} 1^{k_1} & 0 & 0 & 0^n & 0 & \underset{3D'}{w_2} & \dots & w_l & 1^{k_2+1} \end{smallmatrix}\right)$ for (i)

(b) $\left(\begin{smallmatrix} 1^{k_1+1} & w_1 & \dots & \underset{3G'}{w_l} & 0 & 0^n & 1^{k_2+1} \end{smallmatrix}\right)$ for (ii)

Now we apply Corollary 1 in order to reach the next configurations:

- $\left(\begin{smallmatrix} 1 & 0^{k_1+k_2+m} & \underset{3D'}{1} \end{smallmatrix}\right)$ for (a)
- $\left(\begin{smallmatrix} \underset{3G'}{1} & 0^{k_1+k_2+m} & 1 \end{smallmatrix}\right)$ for (b)

□

Lemma 10. $\left(\begin{smallmatrix} 1 & \underset{3G'}{1} & 0^n & 1 & 1 \end{smallmatrix}\right) \vdash^* \left(\begin{smallmatrix} 1 & 0^{n+2} & \underset{3D'}{1} \end{smallmatrix}\right)$.

Proof.

$$\begin{array}{l} \left(\begin{smallmatrix} 1 & \underset{3G'}{1} & 0^n & 1 & 1 \end{smallmatrix}\right) \\ \qquad \vdash^2 \\ \left(\begin{smallmatrix} 1 & \underset{3D}{0} & 0^n & 1 & 1 \end{smallmatrix}\right) \\ \qquad \vdash^{D_3(n)} \\ \left(\begin{smallmatrix} 1 & 0 & 0^n & \underset{3D}{1} & 1 \end{smallmatrix}\right) \\ \qquad \vdash \\ \left(\begin{smallmatrix} 1 & 0 & 0^n & 0 & \underset{3D'}{1} \end{smallmatrix}\right) \\ \qquad \blacksquare \end{array}$$

□

Theorem 2. *The BinSmart machine is minimal.*

Proof. Let w be an arbitrary finite word of length l, $r \in Q$ an arbitrary state, $i \in \{1,2,3,...,l\}$ and $x \in X$ an arbitrary bi-infinite configuration. It is enough to prove that the orbit of x contains the following configurations in the next order:

1. $w = \left({}^{0}_{1D}\ 0^n \right)$
2. $u = \left(1^{k_1+1}\ w_1\ \ldots {}^{w_i}_{r} \ldots\ w_l\ 1^{k_2+1} \right)$
3. $v = \left(1\ 0^m\ {}^{1}_{3D'}\ 0^{n-m-1} \right)$

for any $n, m, k_1, k_2 \in \mathbb{N}$, where $n > m$. Let us see the evolution.

- Evolution from x to w: done directly by Lemma 5.
- Evolution from w to v: done directly by Lemma 6.
- Evolution from u to v: using Lemma 9 and Lemma 10 for $l + k_1 + k_2 = m$
- Evolution from x to u: first of all, note that n is as big as we want, then always exists a path that is longer enough to include u before reach v, even if x contains v, all we have to do is to let it evolve into w with a big enough value of n. Considering this, that both x and u reach v, and the fact that the machine is reversible, there is only one way to reach and specific configuration, we can deduce that x passes through u before evolving into v.

□

Corollary 2. *The BinSmart machine is transitive.*

4 BINSMART'S *T*-SHIFT.

In this section, we prove that the BinSmart's t-shift is a substitutive subshift. First, we first need to define the following recursive functions.

- $S^1_D : \mathbb{N} \to (Q \times \Sigma)^*$
  $S^1_D(0) = {}^{0}_{1D'}\ {}^{1}_{1G}\ {}^{1}_{3D'}$
  $S^1_D(1) = {}^{0}_{1D'}\ {}^{1}_{1G}\ {}^{1}_{3D'}\ {}^{0}_{1D'}\ {}^{0}_{1G}\ {}^{1}_{1G'}\ {}^{1}_{3D}\ {}^{1}_{3D'}$
  $S^1_D(n) = S^1_D(n-1)\ {}^{0}_{1D'}\ {}^{0}_{1G}\ S^1_G(n-2)\ {}^{1}_{1G'}\ S^3_D(n-2)\ {}^{1}_{3D}\ {}^{1}_{3D'}$

- $S^1_G : \mathbb{N} \to (Q \times \Sigma)^*$
  $S^1_G(0) = {}^{0}_{1G'}\ {}^{1}_{1D}\ {}^{1}_{3G'}$
  $S^1_G(1) = {}^{0}_{1G'}\ {}^{1}_{1D}\ {}^{1}_{3G'}\ {}^{0}_{1G'}\ {}^{0}_{1D}\ {}^{1}_{1D'}\ {}^{1}_{3G}\ {}^{1}_{3G'}$
  $S^1_G(n) = S^1_G(n-1)\ {}^{0}_{1G'}\ {}^{0}_{1D}\ S^1_D(n-2)\ {}^{1}_{1D'}\ S^3_G(n-2)\ {}^{1}_{3G}\ {}^{1}_{3G'}$

- $S^3_D : \mathbb{N} \to (Q \times \Sigma)^*$
  $S^3_D(0) = {}^{0}_{3D}\,{}^{1}_{1D}\,{}^{0}_{3G'}$
  $S^3_D(1) = {}^{0}_{3D}\,{}^{0}_{1D}\,{}^{1}_{1D'}\,{}^{1}_{3G}\,{}^{0}_{3G'}\,{}^{0}_{3D}\,{}^{1}_{1D}\,{}^{0}_{3G'}$
  $S^3_D(n) = {}^{0}_{3D}\,{}^{0}_{1D}\,S^1_D(n-2)\,{}^{1}_{1D'}\,S^3_G(n-2)\,{}^{1}_{3G}\,{}^{0}_{3G'}\,S^3_D(n-1)$

- $S^3_G : \mathbb{N} \to (Q \times \Sigma)^*$
  $S^3_G(0) = {}^{0}_{3G}\,{}^{1}_{1G}\,{}^{0}_{3D'}$
  $S^3_G(1) = {}^{0}_{3G}\,{}^{0}_{1G}\,{}^{1}_{1G'}\,{}^{1}_{3D}\,{}^{0}_{3D'}\,{}^{0}_{3G}\,{}^{1}_{1G}\,{}^{0}_{3D'}$
  $S^3_G(n) = {}^{0}_{3G}\,{}^{0}_{1G}\,S^1_G(n-2)\,{}^{1}_{1G'}\,S^3_D(n-2)\,{}^{1}_{3D}\,{}^{0}_{3D'}\,S^3_G(n-1)$

Lemma 11. *$S^1_D(n)$ is the trace corresponding to applying the proposition $D'_1(n)$ to $\left({}^{1}_{}\,{}^{0}_{1D'}\,{}^{0^n}\,{}^{s}\right)$ until $\left({}^{1\,0\,0^n}_{}\,{}^{s}_{1D'}\right)$. The analogous goes for $S^1_G(n)$.*

Similarly, $S^3_D(n)$ is the trace corresponding to applying the proposition $D_3(n)$ to $\left({}^{0}_{3D}\,{}^{0^n}\,{}^{1}\right)$ until $\left({}^{0\,0^n}_{}\,{}^{1}_{3D}\right)$. The analogous goes for $S^3_G(n)$.

Proof. It is enough to see the proof of Lemma 1 and take the trace. □

Now let us define the substitution. Since the states $\{1D, 1D', 3D, 3D'\}$ are symmetrical to the states $\{1G, 1G', 3G, 3G'\}$, we only define the substitution for the first ones.
$\phi : (Q \times \Sigma)^* \to (Q \times \Sigma)^*$
$\phi({}^{0}_{1D}) = {}^{0}_{1D'}\,{}^{1}_{1G}\,{}^{1}_{3D'}$
$\phi({}^{1}_{1D}) = {}^{1}_{1D'}$
$\phi({}^{0}_{1D'}) = {}^{0}_{1D'}\,{}^{0}_{1G}$
$\phi({}^{1}_{1D'}) = {}^{1}_{1D'}$
$\phi({}^{0}_{3D}) = {}^{0}_{3D}\,{}^{0}_{1D}$
$\phi({}^{1}_{3D}) = {}^{0}_{3D}\,{}^{1}_{1D}\,{}^{0}_{3G'}$
$\phi({}^{0}_{3D'}) = {}^{1}_{3D}\,{}^{0}_{3D'}$
$\phi({}^{1}_{3D'}) = {}^{1}_{3D}\,{}^{1}_{3D'}$

For example, the substitution of ${}^{0}_{1G'}\,{}^{1}_{1D}\,{}^{1}_{3G'}$ is:

$$\phi({}^{0}_{1G'}\,{}^{1}_{1D}\,{}^{1}_{3G'}) = {}^{0}_{1G'}\,{}^{0}_{1D}\,{}^{1}_{1D'}\,{}^{1}_{3G}\,{}^{1}_{3G'}$$

Lemma 12. $S^1_D(n) = S^1_D(0)\phi(S^1_D(n-1))$
$S^1_G(n) = S^1_G(0)\phi(S^1_G(n-1))$

$S^3_D(n) = \phi(S^3_D(n-1))S^3_D(0)$
$S^3_G(n) = \phi(S^3_G(n-1))S^3_G(0)$

Proof. It is enough to prove it for $S^1_D(n)$ and $S^3_D(n)$, the other cases can be proved by symmetry.

$$S^1_D(0)\phi(S^1_D(n)) = S^1_D(0)\phi(S^1_D(n-1)\,{}^{0}_{1D'}\,{}^{0}_{1G}S^1_G(n-2)\,{}^{1}_{1G'}\,S^3_D(n-2)\,{}^{1}_{3D}\,{}^{1}_{3D'})$$

$$= S^1_D(0)\phi(S^1_D(n-1))\,{}^{0}_{1D'}\,{}^{0}_{1G}\,{}^{0}_{1G'}\,{}^{1}_{1D}\,{}^{1}_{3G'}\phi(S^1_G(n-2))\,{}^{1}_{1G'}\,\phi(S^3_D(n-2))\,{}^{0}_{3D}\,{}^{1}_{1D}\,{}^{0}_{3G'}\,{}^{1}_{3D}\,{}^{1}_{3D'}$$

$$= S^1_D(0)\phi(S^1_D(n-1))\,{}^{0}_{1D'}\,{}^{0}_{1G}S^1_G(0)\phi(S^1_G(n-2))\,{}^{1}_{1G'}\,\phi(S^3_D(n-2))S^3_D(0)\,{}^{1}_{3D}\,{}^{1}_{3D'}$$

$$= S^1_D(n)\,{}^{0}_{1D'}\,{}^{0}_{1G}S^1_G(n-1)\,{}^{1}_{1G'}\,S^3_D(n-1)\,{}^{1}_{3D}\,{}^{1}_{3D'}$$

$$= S^1_D(n+1)$$

$$\phi(S^3_D(n))S^3_D(0) = \phi({}^{0}_{3D}\,{}^{0}_{1D}S^1_D(n-2)\,{}^{1}_{1D'}\,S^3_G(n-2)\,{}^{1}_{3G}\,{}^{0}_{3G'}\,S^3_D(n-1))S^3_D(0)$$

$$= {}^{0}_{3D}\,{}^{0}_{1D}\,{}^{0}_{1D'}\,{}^{1}_{1G}\,{}^{1}_{3D'}\phi(S^1_D(n-2))\,{}^{1}_{1D'}\,\phi(S^3_G(n-2))\,{}^{0}_{3G}\,{}^{1}_{1G}\,{}^{0}_{3D'}\,{}^{1}_{3G}\,{}^{0}_{3G'}\,\phi(S^3_D(n-1))S^3_D(0)$$

$$= {}^{0}_{3D}\,{}^{0}_{1D}S^1_D(0)\phi(S^1_D(n-2))\,{}^{1}_{1D'}\,\phi(S^3_G(n-2))S^3_G(0)\,{}^{1}_{3G}\,{}^{0}_{3G'}\,\phi(S^3_D(n-1))S^3_D(0)$$

$$= {}^{0}_{3D}\,{}^{0}_{1D}S^1_D(n-1)\,{}^{1}_{1D'}\,S^3_G(n-1)\,{}^{1}_{3G}\,{}^{0}_{3G'}\,S^3_D(n)$$

$$= S^3_D(n+1)$$ □

Theorem 3. *The t-shift of BinSmart is the closure of a fixed point of substitution ϕ.*

Proof. it is enough to prove that $\phi^n({}^{0}_{1G'}) = {}^{0}_{1G'}\,{}^{0}_{1D}S^1_D(n-2)$ for all $n > 1$, because, from Lemma 11, ${}^{0}_{1G'}\,{}^{0}_{1D}S^1_D(n-2)$ is the trace of $\left({}^{0}_{1G'}\,{}^{0^\omega}\right)$ over the first steps and, as the configuration is transitive, the orbit of this configuration is dense. We will prove it by induction.

Base of induction: $\phi^2({}^{0}_{1G'}) = {}^{0}_{1G'}\,{}^{0}_{1D}S^1_D(0)$
Induction hypothesis: $\phi^n({}^{0}_{1G'}) = {}^{0}_{1G'}\,{}^{0}_{1D}S^1_D(n-2)$

Induction thesis:

$$\begin{aligned}
\phi^{n+1}(\,{}^{0}_{1G'}\,) &= \phi(\phi^n(\,{}^{0}_{1G'}\,)) \text{ // Induction hypothesis} \\
&= \phi(\,{}^{0}_{1G'}\,{}^{0}_{1D}\,S^1_D(n-2)) \\
&= {}^{0}_{1G'}\,{}^{0}_{1D}\,{}^{0}_{1D'}\,{}^{1}_{1G}\,{}^{1}_{3D'}\,\phi(S^1_D(n-2)) \\
&= {}^{0}_{1G'}\,{}^{0}_{1D}\,S^1_D(0)\phi(S^1_D(n-2)) \text{ // Lemma 12} \\
&= {}^{0}_{1G'}\,{}^{0}_{1D}\,S^1_D(n-1)
\end{aligned}$$

□

5 ACKNOWLEDGMENTS

The authors want to thank Julien Cassaigne which facilitates the construction of the BinSmart machine. This work was partially financed by ANID/CONICYT + FONDECYT + 11170177.

REFERENCES

[1] Ethan Akin and Jeffrey D Carlson. (2012). Conceptions of topological transitivity. *Topology and its Applications*, 159(12):2815–2830.

[2] Vincent D Blondel, Julien Cassaigne, and Codrin Nichitiu. (2002). On the presence of periodic configurations in Turing machines and in counter machines. *Theoretical Computer Science*, 289(1):573–590.

[3] Julien Cassaigne, Nicolas Ollinger, and Rodrigo Torres-Avilés. (2017). A small minimal aperiodic reversible Turing machine. *Journal of Computer and System Sciences*, 84:288–301.

[4] Anahí Gajardo, Nicolas Ollinger, and Rodrigo Torres-Avilés. (2015). The transitivity problem of Turing machines. In *International Symposium on Mathematical Foundations of Computer Science*, pages 231–242. Springer.

[5] Philip K Hooper. (1966). The undecidability of the Turing machine immortality problem 1. *The Journal of Symbolic Logic*, 31(2):219–234.

[6] E. Jeandel. (2012). On immortal configurations in Turing machines. In S. B. Cooper, A. Dawar, and B. Löwe, editors, *Conference on Computability in Europe (CiE 2012)*, volume 7318 of *Lecture Notes in Computer Science*, pages 334–343. Springer, Springer.

[7] Jarkko Kari and Nicolas Ollinger. (2008). Periodicity and immortality in reversible computing. In *International Symposium on Mathematical Foundations of Computer Science*, pages 419–430. Springer.

[8] Petr Kůrka. (1997). On topological dynamics of Turing machines. *Theoretical Computer Science*, 174(1-2):203–216.

[9] Cristopher Moore. (1990). Unpredictability and undecidability in dynamical systems. *Physical Review Letters*, 64(20):2354.

[10] Rodrigo Torres, Nicolas Ollinger, and Anahi Gajardo. (2012). Undecidability of the surjectivity of the subshift associated to a Turing machine. In *International Workshop on Reversible Computation*, pages 44–56. Springer.

[11] Alan Mathison Turing. (1937). On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London mathematical society*, 2(1):230–265.